Object Indexing Is a Complex Matter*

Bart Lamiroy, Patrick Gros
Movi - Gravir†
655, Avenue de l'Europe
38330 Montbonnot-St. Martin
France
Bart.Lamiroy@imag.fr

Abstract

In this paper we address de problem of algorithmic complexity relative to object recognition through indexing of local descriptors. Indeed, recent work has shown the possible advantages of these recognition techniques which are generally robust and efficient. Although they appear to be potentially very fast, our study shows that the overall complexity of the method is quadratic in the number of local descriptors per image, raising the question of the usefulness of these approaches for large indexing bases of complex images. We show, however, that a careful choice of descriptors may sufficiently reduce the inherent overhead in real applications. As a result we advance that it is more useful to use high-dimensional local descriptors which may be less discriminative, rather than lower-dimensional descriptors with a high expressive value to achieve an optimal recognition result.

Keywords : indexing, recognition, complexity

1 Introduction

Current advances in object recognition seem to be opposing local versus global techniques. Global techniques mainly consisting of considering an image as a whole for comparison with other images [MN95, HLO96], local techniques considering images as a group of local properties [LW88, LG96, SM96]. In the latter case, the local properties are compared in order to differentiate several images. The currently most promising methods of local techniques all rely on a similar underlying scheme. In this paper, we are going to analyze this scheme to understand it forces and weaknesses on a computational basis, in order to achieve optimal performance for this class of recognition methods.

*This paper was published at the *10th Scandinavian Conference on Image Analysis*, Lappeenranta, Finland, June 1997

†Gravir is a joint research programme between the Cnrs, Inpg, Inria Rhône-Alpes and Ujf.

1.1 Underlying Algorithm

Basically all methods, be they local or global, have the same goal :

1. Construct a database of known models. This database is structured differently according to different approaches, but the main objective is to render comparison between an unknown image and a model a relatively easy to compute task. For global methods this consists in applying a transformation to the image in order to simplify its high complexity by either applying statistical or algebraic operators and projecting this in a lower dimensional space[MN95]. An image is a point in this space. Local methods extract a number of regions of interest (their sum being considerably inferior to the whole image by some order of magnitude) on which transformations are performed in order to classify these regions. An image is then defined as a set of region types[SM96]. The transformations are chosen to be invariant to a certain number of operators on the scene, allowing for recognition in a range of situations that differ from the known configuration.

2. At the recognition stage an unknown image is confronted to the constructed model database by having it undergo the same treatment as the initial models. Matching is done by either comparing the distance between points in a moderately high dimensional space for global methods[HLO96], or by comparing sets of low dimensional local region types[LG96].

1.2 Local Methods

We are going to detail a bit further the approach of local methods. The local regions mentioned earlier generally consist of some simple geometric objects in the image, points and lines being the most common. The principle is as follows :

First extract the geometric information from the image. Different methods vary widely, ranging from calcu-

lating highest gradient points to applying complex high texture filters or using polygonal approximation of contour points.

The extracted geometric configurations are then characterized in function of their context within the image. The main idea being the determination of invariant features that would allow to recognize the image in a range of different contexts. We shall call the thus characterized geometric object, a *local descriptor*. Note that this characterization in no way has to be scalar, but is generally represented by some n-dimensional vector. The *local descriptor* generally is invariant to a certain number of transformations on the image. The most common are affine or similarity transforms. This invariance allows the system to recognize similar configurations that are transformed under the given family, and therefore enhances the scope of application without introducing a too complex a modelisation of the indexed objects.

An image is now characterized as a unordered set of *local descriptors*. These descriptors presenting some invariant nature, and all having an identical structure, they can be used as indexing keys in a database containing different images. One has to consider, however, that a given image has several indexing keys, and that a same key can belong to different images.

A reference database is now constructed. It contains the model images to recognize, indexed by their local descriptors in an n-dimensional indexing space.

When an unknown image is presented to the database, it undergoes the same treatment as the model images. It is transformed in a set of *local descriptors* which are checked for in the indexing space. For each descriptor, we get the set of models containing a similar value. As a result we get a list of possible local matches between the image and a series of models. These possible matches are commonly called votes.

In order to determine which of these models most closely resembles the image, a selection criterion has to be applied. Earlier methods [LW88, SM96] simply relied on majority voting. Newer approaches search [LG96] for some more complex global consistency measure between the votes, applying geometrical constraints or introducing statistical information. However, each approach reviews the entire set of votes for some global measure that determines the best model.

Since this recognition scheme has shown to be potentially applicable in a large range of situations, we are going to give its underlying complexity a closer look.

2 Complexity

To summarize, recognition is done in two steps : matching of *local descriptors* by simply retrieving model instances through indexing, and hypothesis verification through a global consistency measure on the found matches. In this section we shall first express the number of found matches of the first step, and conclude by calculating the global complexity of the recognition algorithm in a second stage.

2.1 Index Space Size

Consider an n-dimensional indexing table containing *local descriptors* of M models. We shall assume that a model is defined by $\tilde{D}$ descriptors on average. We should note that the number of descriptors D in an image principally depends on the type of images considered for indexing. E.g. if the *local descriptors* are based on high texture interest points, D will typically be high for textured images and low for line drawings for instance. Since most indexing applications treat similar kinds of images, we can assume that D does not vary too much from one image to another. For ease of use we can therefore consider that $D \simeq \tilde{D}$ and constant over all images of the given application. The number of *local descriptors* in the database (or indexing table) is $\tilde{D}.M$.

Let k_i be the subdivision of the indexing table in dimension $1 \leq i \leq n$. Then, the average number of *local descriptors* in one indexing bin $\bar{I}$ is therefore

$$\bar{I} = \frac{\tilde{D}.M}{\prod_{i=1}^{n} k_i} \tag{1}$$

Let us now apply this result to the generic recognition algorithm. As we have seen in 1.2, the usual process is to take an unknown image and measure it against the indexed model base. As shown in the previous paragraph, we can safely consider that this new image also contains $\tilde{D}$ descriptors.

At first, we are going to consider noiseless matching . In a second step we shall show how the introduction of noise affects global complexity.

2.2 Noiseless Matching

In this case, *local descriptors* have an exact value. The part of the algorithm that consists in selecting the plausible local matches between the unknown image and the different indexed models therefore consists of the following loop :

```
Input Arguments : IMAGE, MODELBASE
Return Argument : LIST<I_descr, M_descr, M>

begin
    OUTPUTLIST = empty match list;

    for all descriptors D in IMAGE do
    begin
        get INDEX in MODELBASE corresponding
            to D;
```

```
        get list of model descriptors at
            INDEX in MODELBASE and append
            the result to OUTPUTLIST;
    end;

    return OUTPUTLIST;
end;
```

In above algorithm, we go, on average $\tilde{D}$ times through the loop, accessing an indexing bin at each iteration. As seen in equation (1) $\bar{I}$ matches are added to the output list at each step. Therefore, the output of this routine is a list of $\tilde{D}.\bar{I}$, or in a more detailed form,

$$\frac{\tilde{D}^2.M}{\prod_{i=i}^{n} k_i} \tag{2}$$

model descriptors on average.

2.3 Matching with Noise

We are now considering the more realistic case where *local descriptors* are bound to noise. In the most general case we can assume that the noise is Gaussian and that the different "dimensions" may be correlated. *Local descriptors* should therefore be compared up to an n-dimensional ellipsoïd to the model descriptors. This ellipsoïd is defined by the covariance matrix. From an indexing point of view, this corresponds roughly to consider not only the descriptors located at a given index, but also to take into account its neighbors. In what follows we are going to assume that the correlation matrix is diagonal (i.e. there exists no correlation between the different dimensions of the descriptors). This assumption can be taken since the covariance matrix is a positive defined matrix, and can therefore be diagonalized if necessary. From a probabilistic point of view, this corresponds to operate a principal components analysis and re-express the data in an appropriate reference frame.

The axes of the ellipsoïd are now parallel to those of the reference frame of the descriptors. In order to recover the area spanned by a descriptor and its attached uncertainty in the indexing space, we can approximate it by a kind of hypercube centered in the index corresponding to the considered descriptor. Let $\mathcal{N}(\varepsilon_i)$ be the neighborhood around this index in dimension i. In other terms ε_i corresponds to the number of bins in the i^{th} indexing dimension now considered for matching. To retrieve all possible descriptors in this direction we need to access each of these bins.

In comparison to the previous case, we are now going to examine ε_i times more bins for each dimension i. The total number of bins within the given uncertainty ellipsoïd therefore becomes $\prod_{i=i}^{n} \varepsilon_i$. By rearranging to the algorithm shown above in order to consider the neighborhood rather than the unique index, we obtain that the output list for the case of noisy descriptors contains

$$\left(\prod_{i=i}^{n} \varepsilon_i\right) \cdot \frac{\tilde{D}^2.M}{\prod_{i=i}^{n} k_i}$$

model descriptors, which can be rewritten as

$$\frac{\tilde{D}^2.M}{\prod_{i=i}^{n} \frac{k_i}{\varepsilon_i}} \tag{3}$$

2.4 Global Recognition Complexity

As pointed out, the recognition algorithm consists of a matching phase and a global verification phase. The complexity of the first part is simply the complexity of accessing an index space, the second phase depends of the output of the first. We shall call this output S_m.

The global complexity for recognition of a given image $\mathcal{I}$ therefore becomes $D_{\mathcal{I}}.\mathcal{C}_a + \mathcal{C}_v(S_m)$ where

$D_{\mathcal{I}}$	corresponds to the number of *local descriptors* in $\mathcal{I}$
$\mathcal{C}_a$	corresponds to the complexity of accessing an index in the indexing space
$\mathcal{C}_v(S_m)$	corresponds to the complexity of applying a global consistency check on S_m matches.

Minimum average complexity for $\mathcal{C}_a$ is clearly $\mathcal{O}(1)$. We can also suppose that $D_{\mathcal{I}} = \tilde{D}$, while equation (3), gives us the value for S_m. Remains the complexity for the global consistency check $\mathcal{C}_v(S_m)$. We are going to show that on average this is $\mathcal{O}(S_m)$ at best.

At the first stage of the algorithm, we recover an unsorted set of plausible matches between *local descriptors* of the image with those of several models. The aim is to sort these models in function of the achieved matches, best model first. In order to sort the models of the list, each of them has to be given a weight. Since there is no ranked order between the found matches with a given model, each match concerning this model has to be taken into account for determining the weight of the latter. Therefore, we conclude that, to establish a sorted list of the plausible models, every single match has to be taken into account, leading to the final complexity of $\mathcal{O}(S_m)$.

As a conclusion the recognition complexity is given by

$$\tilde{D} + \frac{\tilde{D}^2.M}{\prod_{i=i}^{n} \frac{k_i}{\varepsilon_i}}$$

which, asymptotically is equivalent to

$$\frac{\tilde{D}^2.M}{\prod_{i=i}^{n} \frac{k_i}{\varepsilon_i}} \tag{4}$$

3 Analysis

In this section we are going to analyze the expression in equation (4) in order to achieve a better understanding of the different parameters that influence on the recognition performances. Experiments with given recognition models have shown that the influence of the number of descriptors in an image has a severe limiting effect in the case of real world images [LSM+96]. This analysis will show how to avoid this problem. In order to present some results as an illustration, we used the indexing-recognition method described in [LG96]. *"Local descriptors"* are calculated on configurations of three or four points in a segmented image that are connected by edge segments. They consist of the angle α and length ratio ρ between connected segments taken two by two, and, optionally, the orientation of the segments, calculated from the original grey level image. This makes the size of the descriptor vary from 2 dimensions in the simple 3-point case (α, ρ) to 5 dimensions in the 4-point case $(\alpha_{123}, \alpha_{234}, \rho_{123}, \rho_{234}, c)$ (where the subscripts denote the indexes of the points on which the value is calculated, and c a discrete code describing the mutual orientation of the segments). To give an idea of numbers TABLE 1 gives some typical values for two types of used images. FIG. 3 and FIG. 4 give an idea of the kind of images used.

Configuration	Image Points	Descriptors
3-Point	62	113
4-Point	62	215
3-Point	922	1604
4-Point	922	4038

Table 1: Some typical values for descriptors

In the following sections execution times are in seconds, and given for comparison purposes only. They correspond to non optimized code, executed on different machines. We index angles in a range $[0, 180]$ in $20°$ bins, and ratios from $\frac{1}{60}$ to 60 (using log) in log(9) bins. For each dimension (with exception of orientation) ε_i is 2. These values are ad hoc, and in no way intended to be optimal, or to be taken as important parameters. They are given as illustration of the theoretical values.

3.1 Influence of the Size of the Database

The size of the database is expressed by M, being the number of models. Few things can be said about the influence of M on global performance. In the previous section we expressed the complexity for sorting all models by rank. However, in most cases the main interest goes to the highest ranked model. In other terms, we want

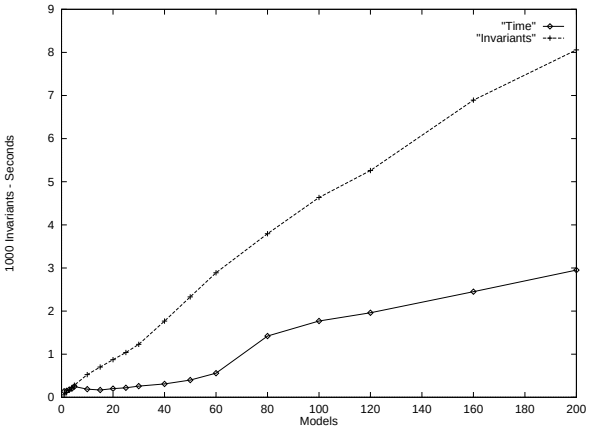

Figure 1: Linear evolution of execution time and bin population with size of the database (in number of models)

to compare our image to the models in order to retrieve the closes model only. Classical complexity studies tell us that the optimal complexity for comparing one element to n is $\mathcal{O}(\log(n))$. We can indeed improve the complexity in M by pre-arranging our models by calculating their relative distance to one another (provided that such a distance exists and is computationally feasible) and by using a selective voting scheme, based on the results obtained by the votes for other models, thus reducing the overall average complexity to $\mathcal{O}(\log(M))$. However, since M is not, by far, the predominant factor, this is of less concern to us. It may be useful however, once the other optimizations have been done, and the aim is to address an as large as possible database.

As an example, we took 200 similar images and progressively indexed them in a database. At each step we interrogated the database with one identical test image, and observed the response time of our indexing system. As one can see in FIG. 1, the execution time varies linearly with the number of models when the models have a similar number of invariants each.

3.2 Influence of the Image Complexity

Of far greater influence than the number of models in the database, is the average complexity $\tilde{D}$ of the images that are considered for indexing and recognition. If one considers an existing recognition model, using some given type of *local descriptors* [SM96, LG96, SC96] the range of application is limited by the complexity of the images only. However, since this complexity is quadratic, the usefulness of these methods seems severely restricted.

As an illustration, TABLE 2 shows recognition times for different kinds of images and for different numbers of indexed models. Although the number of models is not high enough to guarantee a uniform distribution in the indexing space, it can be clearly seen that the evolution

Models	$\tilde{D}_1$	$\tilde{D}_2$	$\tilde{D}_3$
1	8.35	0.2	0.03
2	16.52	0.22	0.03
3	25.33	0.26	0.04
4	35.55	0.35	0.04
16	–	0.95	0.13

Table 2: Execution times for different Image complexities. Average number of descriptors are $\tilde{D}_1 = 3400$, $\tilde{D}_2 = 200$, $\tilde{D}_3 = 55$.

of complexity is towards a quadratic behaviour. We're currently working on the elaboration of more complete sets of databases, in order to confront experimental results to the theory.

At this point it my be useful to address an optimization criterion that one could be tempted to apply. At the sight of the current results it seems obvious that the number of descriptors in an image should be kept to a minimum for the algorithm to behave in a computationally controllable way. What does this mean ?

As per definition of "local" and "global" descriptors, the less numerous descriptors are needed to describe an image, the more "global" these descriptors are. One has to be aware that by modifying the descriptors to obtain a more compact description in terms of the latter, the description becomes more "global" and therefore subject to known difficulties in object recognition as occlusion, change of background, multiple objects in the same scene, etc. Basically, as the description becomes more "global", the method becomes more and more sensible to smaller and smaller differences between different images. As a result, more models are needed in order to achieve the same recognition results as with the more "local" descriptors. In certain cases, this may result in M increasing more than $\tilde{D}^2$ is decreasing, leading to a loss of performance.

3.3 Influence of the Descriptor Size

The most important role belongs to the descriptor size n. It is indeed the only factor that can contribute to reduce the cost of the algorithm. Rather than trying to reduce the number of *local descriptors*, which may lead to a number of inconvenients presented earlier, it is far more interesting to increase the information stored in the descriptors themselves. As an example FIG. 2 shows the cost of the algorithm for different values of $\tilde{D}$ and n.

It is clear that cost decrease is steeper in the direction of n than it is in the direction of $\tilde{D}$.

Another advantage of increasing the dimensionality of the descriptors is that existing modeling remains valid. I.e. there is no need to fundamentally change the modeling as was necessary with the previous solution which consisted in reducing the number of descriptors.

It is not very easy to give a representative image of the change of descriptor size change due to the fact that changing the size of the descriptor often implies a modification in the number of *"local descriptors"* thus changing the overall complexity in a more complex way. In TABLE 3 we're giving the execution times encountered with our system for different cases. Note that passing from simple to oriented (i.e. taking into account the orientation of the segments that form the initial configuration) subdivides the indexing space in 4 in the 3-point case, while it subdivides in 8 for the 4-point case. Average size of indexed and query images corresponds to the values given in TABLE 1.

Configuration Type	Size	Simple Images	Complex Images
3-Point (simple)	2	3.3	235.93
3-Point (oriented)	3	1.18	97.56
4-Point (simple)	4	1.39	130.13
4-Point (oriented)	5	0.43	26.34

Table 3: Influence of descriptor size

3.4 A few Words on Indexing

A corrolary result of the conclusion in 3.3 is that there is no need to be very precise where the indexing bins are concerned, as long as it is possible to preserve a high enough number of dimensions. For instance, it is less advantageous to use 4 dimensions, each subdivided in 100 bins ($n = 4$, $k_i = 100 \forall i$) than to have 9 dimensions subdivided in 10 bins ($n = 9$, $k_i = 10 \forall i$).

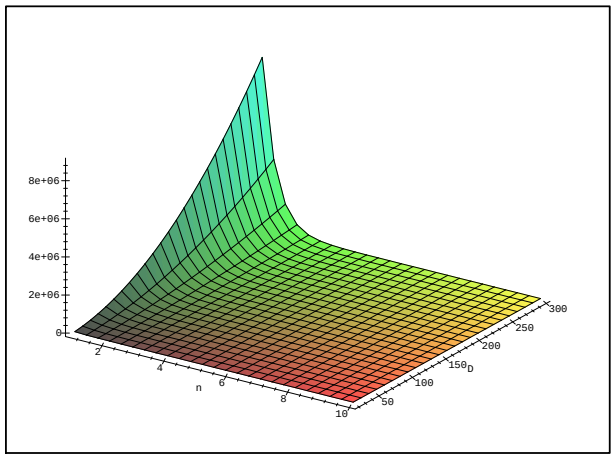

Figure 2: $\tilde{D} \in [20..300]$, $n \in [1..10]$, $k_i = 10$ over all dimensions and $M = 1000$

Although this might seem trivial, it has far stretched results. As the local descriptors are either full invariants, and therefore usually very instable numerically, or either quasi-invariants, and therefore not constant on the whole range of transformations one would like to observe. The possibility of being less stringent when comparing them results in encompassing a broader range of recognizable transformations. It reduces therefore the need of a high number of different models.

As a matter of fact, being less stringent boils down to becoming more qualitative and less quantitative. In a sense this is what recognition is supposed to be, and results in this direction have shown that this is quite a promising way to go [Car96].

4 Conclusion

The complexity study of local indexing techniques shows that for a given level if image complexity $\tilde{D}$ there is a way of reducing the cost of recognition in such a way that these techniques can easily be exploited with a reasonable execution time, provided that the local descriptors used, are of a sufficient high dimension. We have shown that a policy of cost reduction should focus on this dimensionality rather than on the number of descriptors.

Current ongoing work lies for one part in the study of population repartitioning in the database. This will allow us to examine the influence of non-uniform distributions on complexity and recognition. For an other part it consists of measuring the influence of bin sizes on recognition results, as described in section 3.4.

The scope of this paper was not to provide methods of expanding the number of dimensions for a given modelisation as there exist in current know techniques. This also is part of our ongoing research.

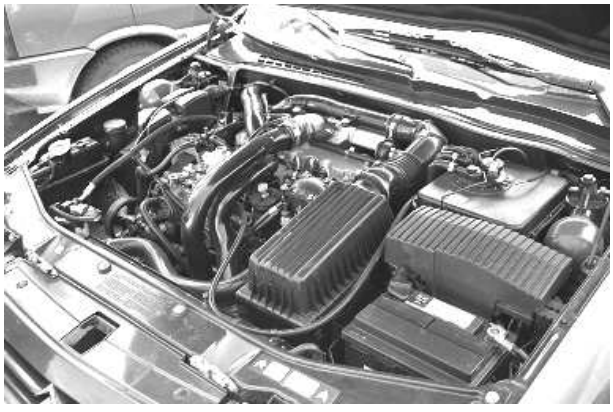

Figure 3: Example of used complex greylevel image

References

[Car96] S. Carlsson. Combinatorial geometry for shape representation and indexing. In J. Ponce, A. Zisserman, and M. Hebert, editors, *Proceedings of the ECCV'96 International Workshop on Object Representation in Computer Vision II, Cambridge, England*, Lecture Notes in Computer Science, pages 53–78. Springer-Verlag, April 1996.

[HLO96] D.P. Huttenlocher, R.H. Lilien, and C.F. Olson. Object recognition using subspace methods. In B. Buxton and R. Cipolla, editors, *Proceedings of the 4th European Conference on Computer Vision, Cambridge, England*, pages 536–545. Springer-Verlag, April 1996.

[LG96] B. Lamiroy and P. Gros. Rapid object indexing and recognition using enhanced geometric hashing. In *Proceedings of the 4th*

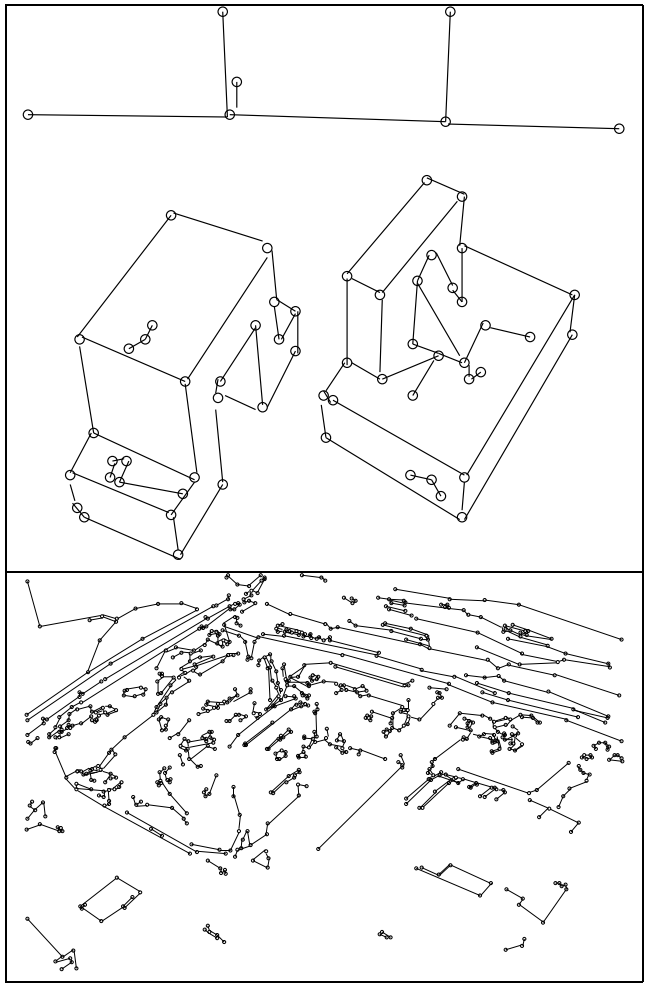

Figure 4: Examples of used segmented images

European Conference on Computer Vision, Cambridge, England, volume 1, pages 59–70, April 1996. Postscript version available by `ftp`[1].

[LSM+96] B. Lamiroy, C. Schmid, R. Mohr, M. Tonko, K. Schäfer, and H.-H. Nagel. Computer aided (dis)assembly using visual cues. In *Proceedings of the I.A.R. Annual Meeting*. I.A.R. (Institut franco-allemand pour les applications de la recherche/ Deutch-Französisches Institut für Automation und Robotik), November 1996. Postscript version available by `ftp`[2].

[LW88] Y. Lamdan and H.J. Wolfson. Geometric hashing: a general and efficient model-based recognition scheme. In *Proceedings of the 2nd International Conference on Computer Vision, Tampa, Florida, USA*, pages 238–249, 1988.

[MN95] H. Murase and S.K. Nayar. Visual learning and recognition of 3D objects from appearance. *International Journal of Computer Vision*, 14:5–24, 1995.

[SC96] B. Schiele and J.L. Crowley. Object recognition using multidimensional receptive field histograms. In *Proceedings of the 4th European Conference on Computer Vision, Cambridge, England*, pages 610–619, 1996.

[SM96] C. Schmid and R. Mohr. Combining grey-value invariants with local constraints for object recognition. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, San Francisco, California, USA*, June 1996. [3]

[1] ftp://ftp.imag.fr/pub/MOVI/publications/Lamiroy_eccv96.ps.gz
[2] ftp://ftp.imag.fr/pub/MOVI/publications/Lamiroy_iar96.ps.gz
[3] ftp://ftp.imag.fr/pub/MOVI/publications/Schmid_cvpr96.ps.gz.